

マイクロコンピュータインタフェースの システム開発について

Design for Micro Computer Interface System

渋 井 二三男*

SHIBUI, Fumio

概要：情報・通信システム開発において、日本のエンジニアが開発できるテリトリーは入力制御部分のインタフェースといっても言い過ぎではない。何故なら、実務経験のある方なら理解できると思うが、①ほとんどのマイクロプロセッサはアメリカ製品で標準製品化されて、②周辺系は富士通ファナックなどのロボットメーカー、YEDaTa 社などの I/O メーカー、日立・NEC などの通信メーカーなどで寡占化、特許化していて、新規参入は至難の技である。そこで、社会的にニーズも高く、ニッチ産業でもある入力制御部分のインタフェースのシステム開発事例について次に説明する。

1. 試作標準インターフェースの概要

本装置は、CPU に Intel 社製、メモリ (ROM&RAM) および周辺インターフェースに PIO (割込みインターフェース)、PIT (タイマ制御インターフェース)、PIO (パラレル入出力制御インターフェース)、SIO (シリアル入出力制御インターフェース) 等により構成され、データバス、アドレスバス、コントロールバスを中心にパネル面に展開表示される。また、各種周辺インターフェースの動作、割込み制御の動作、入出力制御の動作等のインターフェース技術を各種表示モニタ、およびプログラミングにより習得する学習支援システム (以下。システムと略す) を試作し、システムの概略図を次に示す。

2. システムの各仕様

システムの各仕様を次に示す。

(1) CPU

ア. Intel 社製 CPU

イ. データバス、アドレスバス各 8 ビット * N のデータ表示 (入出力動作時のみ)

(2) メモリ

ア. ROM256k バイト (MAX512k バイト実装可)

* 城西大学女子短期大学部

イ. RAM64k バイト

(3) 周辺インターフェース

① 割込み制御インターフェース (PIC : 8259 × 1)

- ア. 割込入力信号数 $8 * N$ 入力
- イ. 手動割り込み発生回路 × 1 回路
- ウ. 割込みマスクレジスタ表示 $8 \text{ ビット} * N$

② タイマ制御インターフェース (PIT : 8253 × 1)

- ア. タイマ数 3 タイマ

③ パラレル入出力制御インターフェース

(PIO : 8255 × 1)

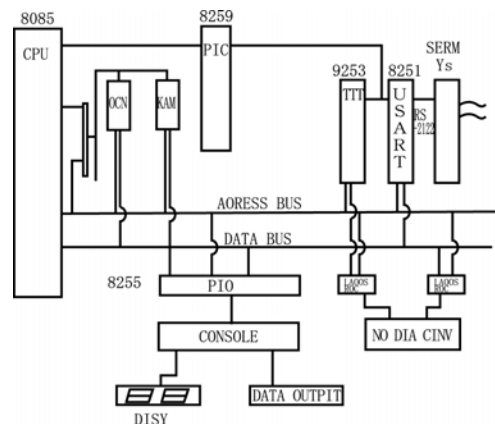
- ア. 数値表示器 × 2 回路
- イ. 8 ビットスイッチレジスタ × 8 回路
- ウ. リレー制御回路 × 8 回路
- エ. プリンタ制御回路 × 1 回路 (セントロニクス仕様プリンタ接続可)

④ シリアル入出力制御インターフェース (SIO : 8251 × 1)

- ア. RS-232 × 1 チャンネル
- イ. モデム制御信号表示 6 ビット

⑤ A/D, D/A 変換インターフェース

- ア. アナログ電圧発生回路 × 3 回路
- イ. デジタルデータ表示 $8 \text{ ビット} * N$



2. 入出力制御方式

ここでは、コンピュータとメモリ、入出力装置との間で、データのやり取り（授受）を行う方法について考えてみる。

コンピュータの動作について簡単にいうと、コンピュータは 10 個程度のステートで 1 つの命令を実行（命令により、若干異なりますが）している。

クロック信号が、2.5MHz とすると、1 ステートの時間は、400ns となる。

したがって、1 つの命令は $400\text{ns} \times 10 = 4000\text{ns}$ ($4 \mu\text{s}$) の時間で実行される。

これで、コンピュータはどのくらいの速さで処理されているか理解できる。

では次に、メモリや入出力装置はどのくらいの速さで動作するのか説明する。

メモリはコンピュータとほぼ同じくらいの速さで動作する。

実はメモリにも速いものと遅いものがある。

→メモリの場合、コンピュータから****番地のデータを下さい！といわれてから、その番地のデータを読み出してくるまでの時間をアクセスタイム (Access Time) といつて、約 50ns～500ns くらいの時間がかかる。

→したがって、計算機より遅いメモリがある。

ところがマイクロコンピュータもうまくできていて、ステート・S2 とステート・S3 の間に、ウェイトステート・Sw という特殊なステート (時間待ちをするステート) を追加できるようになっており、コンピュータより多少遅いものは、このウェイトステートにより救うことができる。

次に出力装置を説明する。

例えば、プリンタを考えてみる。機種により異なるが、昔のタイプライタなら 1 秒間に 10 文字、パソコンなどによく使用されている最近のプリンタで 1 秒間に約数 10～200 文字程度である。

説明上、1 秒間に 100 文字印字できるプリンタとする。

もう一度コンピュータに戻って、ではコンピュータはプリンタに 1 秒間にどのくらいの文字を送信できるか？

これはプログラムにより差があるが、先ほど 1 命令が約 $4\mu s$ くらいで実行されると説明したが、プリンタへデータ (文字) を出力するのに 10 命令前後は必要とするので、1 文字出力するのに約 $40\mu s$ となり、1 秒間に約 25000 文字出力できることになる。

つまり、コンピュータは 25000 文字出力できるのに、プリンタは 100 文字しか印字できないのである。

これでは、お互いがかってに動作しているだけで、正しくプリンタに印字させることができない。

このようにマイコンシステムでは、コンピュータとデータ授受を行うとき、メモリはあまり問題にはならないが、入出力装置の場合は問題が発生する。このため、コンピュータと入出力装置との間でデータ授受をうまく行わせる方法として、以下に示す入出力制御方式がある。

- ・ハードウェアによる制御方式
- ・ソフトウェアによる制御方式 (ポーリング方式)
- ・割込み制御方式
- ・DMA制御方式

(1) ハードウェアによる制御方式

この方法が一番簡単な方法で、マイクロコンピュータのウェイトステートを利用して、コンピュータの速度を入出力装置の速度と同じようにしようという方式である。

図 1.1 にこの方式でのコンピュータとプリンタの動作を示す。

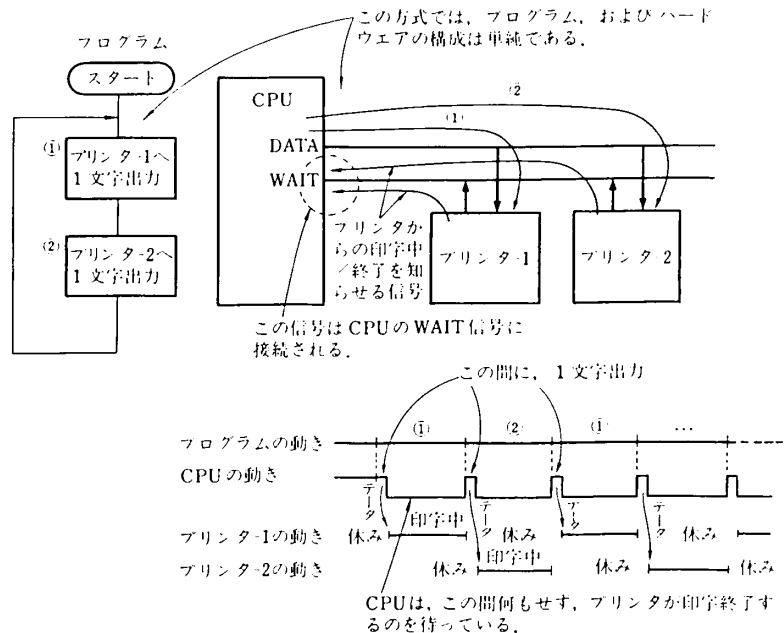


図 1.1 ハードウェアにおける制御方式での動作

この方式では、プログラム、およびハードウェアの構成は単純になるが、CPUの動きはほとんど何もせず、プリンタが印字終了するのを待っているだけである。そのうえ、プリンタにしても1文字印字したら休み、また印字というように、プリンタの印字速度の半分でしか印字していない（さらに、プリンタの数を増やして考えると、もっと効率が悪くなります）ことになる。

したがってこの方式では、CPUの速度をいくら早くしても、プリンタなど、外部装置の速度が高速にならないと効果が上がらない。

いいかえると、CPUと同程度の外部装置であれば、プログラム、ハードウェアの構成が簡単なものだから良い方式であるといえる。したがって、この方式はCPUの速度に比べ、前述したアクセスタイムの遅いメモリなどと、データ授受を行う場合に使用される。

(2) ポーリング方式

この方法も(1)と同じように簡単な方法であるが、(1)はハードウェアで制御したが、ポーリング方式はソフトウェア（プログラム）で制御する方法です。したがって、マイクロコンピュータは(1)のような時間待ちはせず、常に動作している。

どんな方法かというと、コンピュータはプリンタへ1文字出力した後、次のデータを出力する前にプリンタに対して、“さっきのデータの印字は終了しましたか？”とたずね、終了していたら次のデータを出力し、終了していなかったらまたたずねる、という処理をポーリング方式という。

図 1.2 に、この方式でのコンピュータとプリンタの動作を示す。

この方式では(1)のハードウェア制御方式と比べると、プログラム、ハードウェア構成において少し複雑になるが、CPUは印字終了を待っているだけではなくプログラムの指示に従ってプリンタの状態を調べている。

したがって、プリンタ-1, 2 が印字終了したら、ただちに次のデータを出力することができるので、プリンタを効率よく使用することができる。

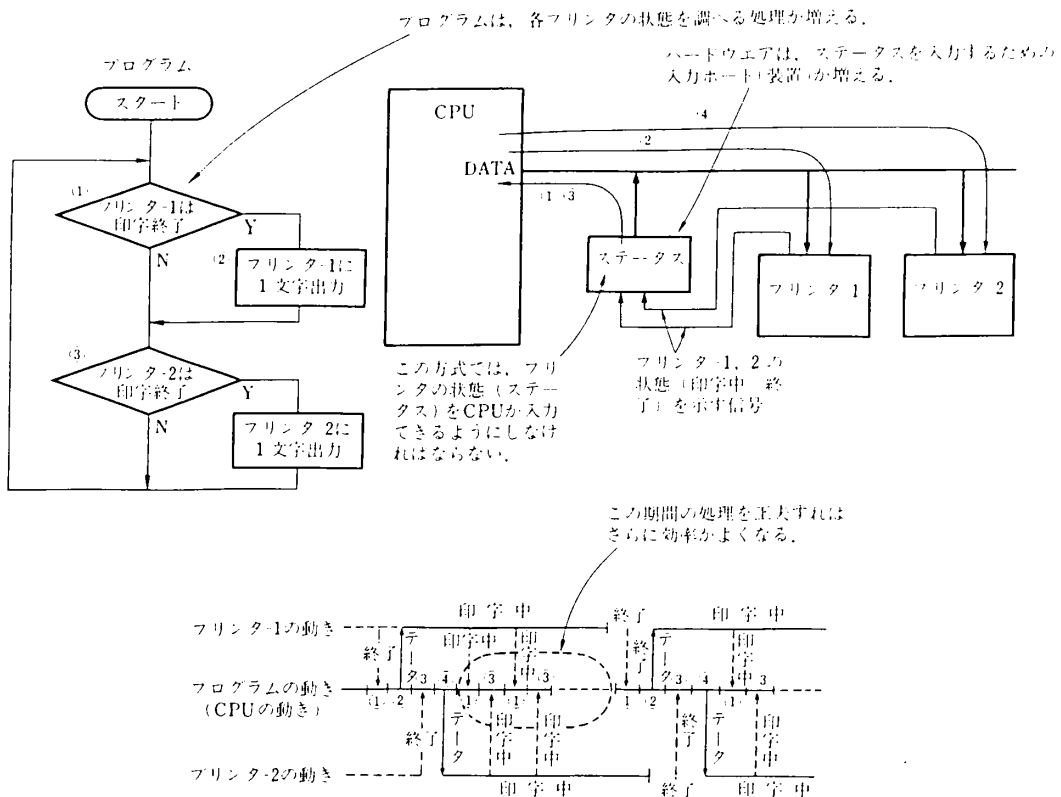


図 1-2 ポーリング方式での動作

さらに、この方式での利点は図 1.3 に示すように、プリンタ-1, 2 が印字している間に、他の処理を実行することができるということである。

マイコンシステムの入出力装置には、今、例に上げているプリンタの他に、いろいろな入出力装置が使用されている。したがって、このポーリング方式では、コンピュータに他の装置の状態(出

力装置ならデータを出力してもよいかなど)をチェックさせ、他の仕事をやらせることができることになる。

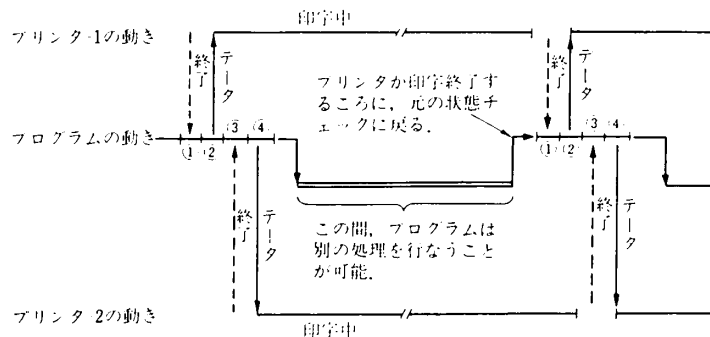


図 1-3 ポーリング方式の応用

(3) 割り込み制御方式

この割り込み制御方式は、(2)のポーリング方式をもう少し改良した方法で、マイクロコンピュータはプリンタへデータ(文字)を出力した後、他の処理に専念する。この後の処理が若干異なり、(2)のポーリング方式の場合は、コンピュータ(プログラム)が、適当な時間(印字が終了する時間)をみはからってプリンタの状態を調べに行ったが、この方式では、プリンタが印字を終了した時点でコンピュータに終了を知らせる。

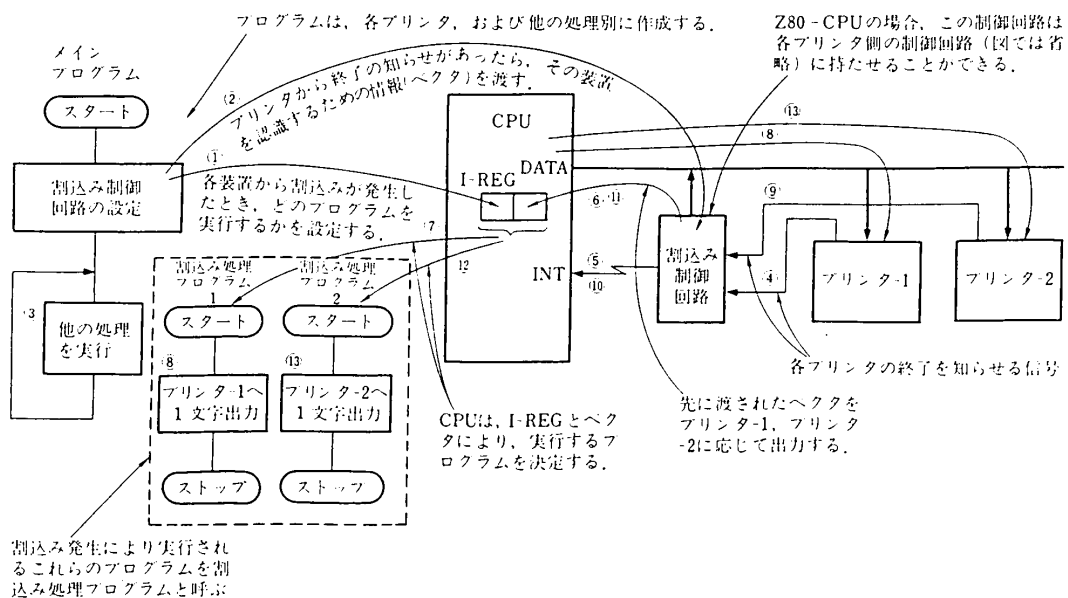


図 1.4-1 割り込み制御方式での動作

この知らせは、割り込み (Interrupt) という機能を使って行われる。コンピュータはこの割り込みが

入ると、今行っていた仕事を一時中断して、プリンタへ次のデータを出力し、また仕事を再開することができ、(2)と比べコンピュータの仕事を効率化させることができる。

図 1-4-1、図 1-4-2 に、この方式でのコンピュータとプリンタの動作を示す。

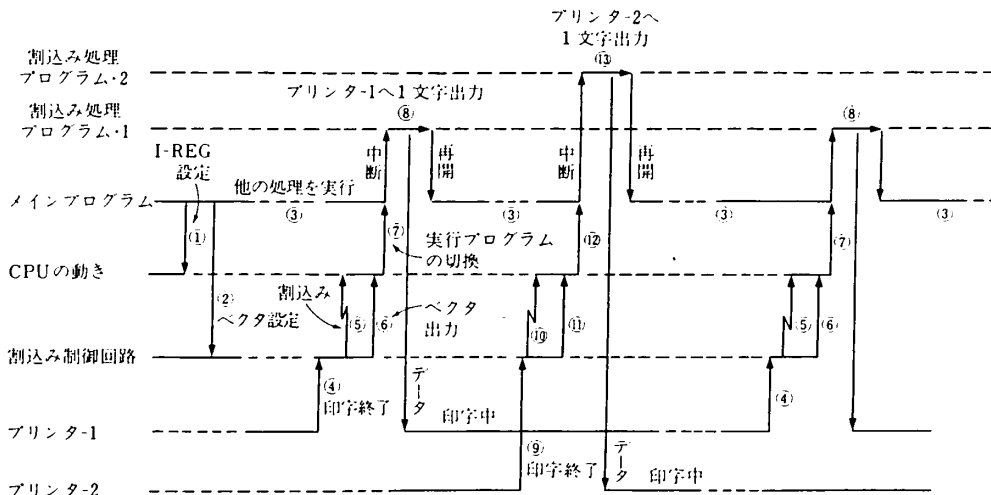


図 1.4-2 割り込み制御方式での動作

(4) DMA制御方式

この方法は、(3)の方式をさらに改良した方法といえる。

DMA (Direct Memory Access) 制御方式は、DMA コントローラにプリンタへの文字転送（出力）を完全にまかせてしまう方法である。

この方法だと、最初にコンピュータがDMA コントローラに対し、メモリ装置のどの番地（転送アドレスという）にあるデータからいくつ（転送数という）をプリンタへ渡せ（メモリからプリンタへ出力するのでメモリ - IO 転送という）という指示をすれば、後はDMA コントローラが(2)，(3)でコンピュータが行っていた処理を代行して、プリンタへ文字を出力する。

この方式もプリンタを例に説明したが、実際はプリンタに使用されることはほとんどない。

では、どのような装置に使用されるかという私たちの身の回りでは、パソコンのフロッピーディスク装置などに使用されている。

フロッピーディスク装置の場合、詳しい説明は省略するが、媒体が、機械的に回転しているため、コンピュータとフロッピーディスク装置間で、データ転送をしようとすると約 $16\mu s \sim 32\mu s$ の間に、1 データの転送を終了させなければならない。

そうすると、約 4～8 命令しか実行できず、これでは、データを次々と読み出してメモリに格納したり、終了判定などはできない。

そのため、この方式では、命令を実行してデータ転送を行うのではなく、ハード的にデータ転送

させているのである。

もう少し詳しく説明すると、マシンサイクルは、CPUがメモリからプログラムを読み出して解読後、その命令を実行していたが、このマシンサイクルをデータをメモリから読み出し外部装置へ出力したり、外部装置から入力したデータをメモリへ書き込むというように変更すると、1 データは1マシンサイクル程度の時間で転送できることになり、時間でみると約 $1.2\mu s \sim 2.4\mu s$ くらいで終了し、前述のフロッピーディスク装置ともデータ転送が可能になる。

DMAコントローラとは、このような処理を内部で行っている。

図 1.5 に、この方式での動作を示す。

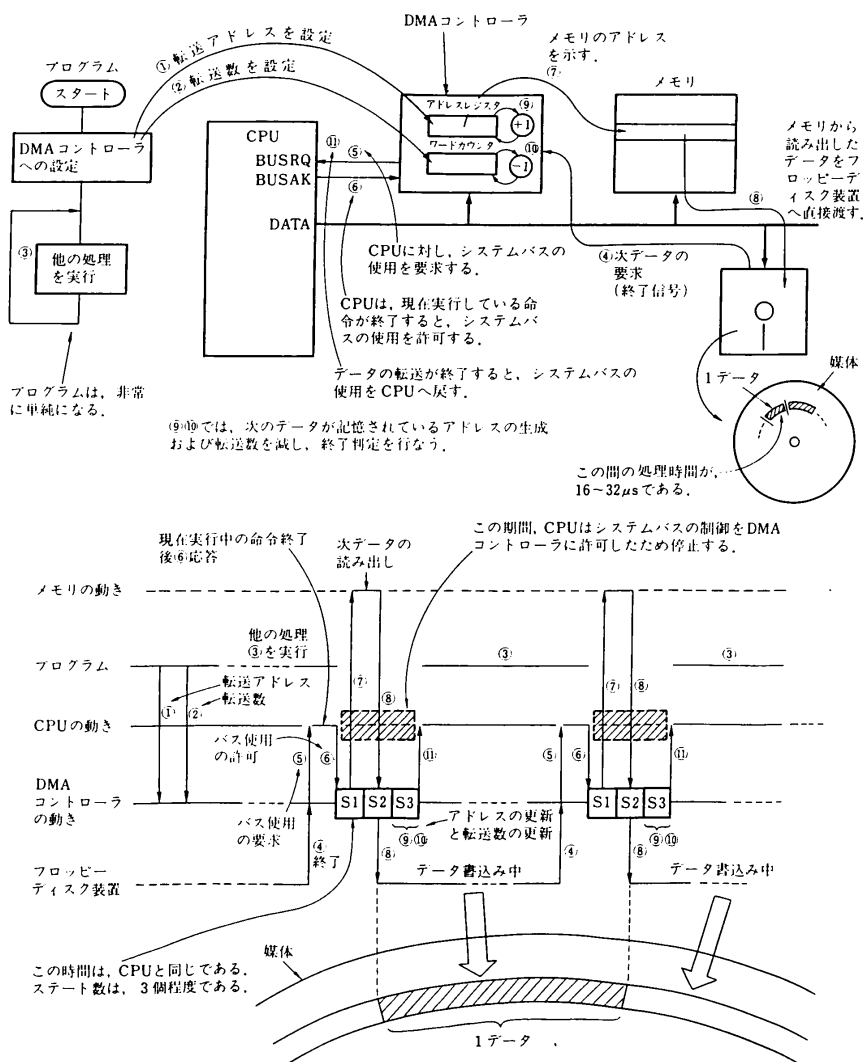


図 1.5 DMA制御方式の動作

マイクロコンピュータシステムでは、以上の4種類の中から1種類を選択して使用するのではな

く、メモリ、入出力装置により、それぞれ使い分けている。したがって、どのようなシステムでも、上記の複数の制御方式が採用されている。

謝 辞

本稿を執筆するにあたり、(有)テクノランド代表取締役社長森田博氏に多大な御指導をいただきました。ここに深く感謝申し上げます。

引用・参考文献

渋井二三男・森田博著：マイコン・トレーニング 初歩から応用まで；日刊工業新聞社

(Received Mar. 31, 2005)